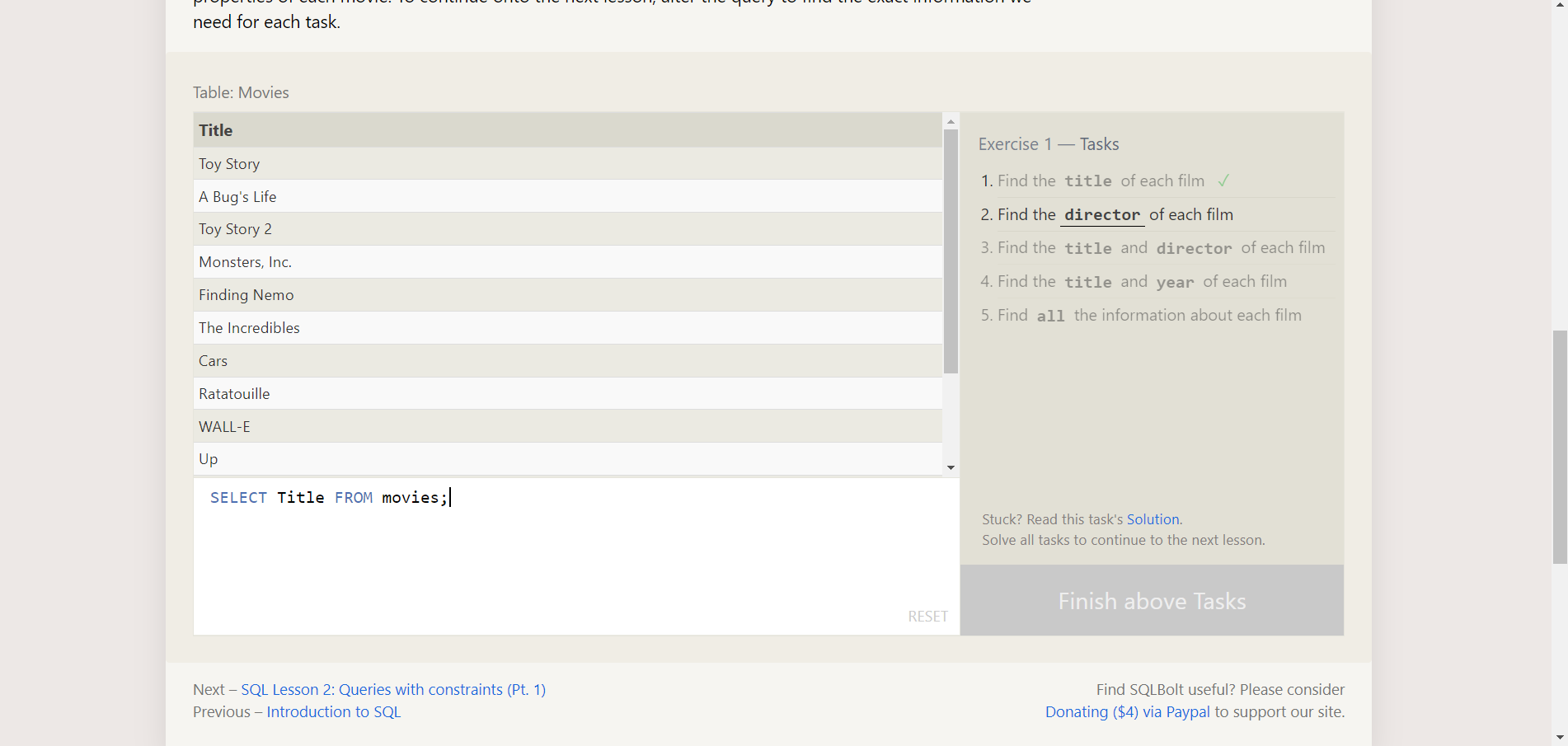
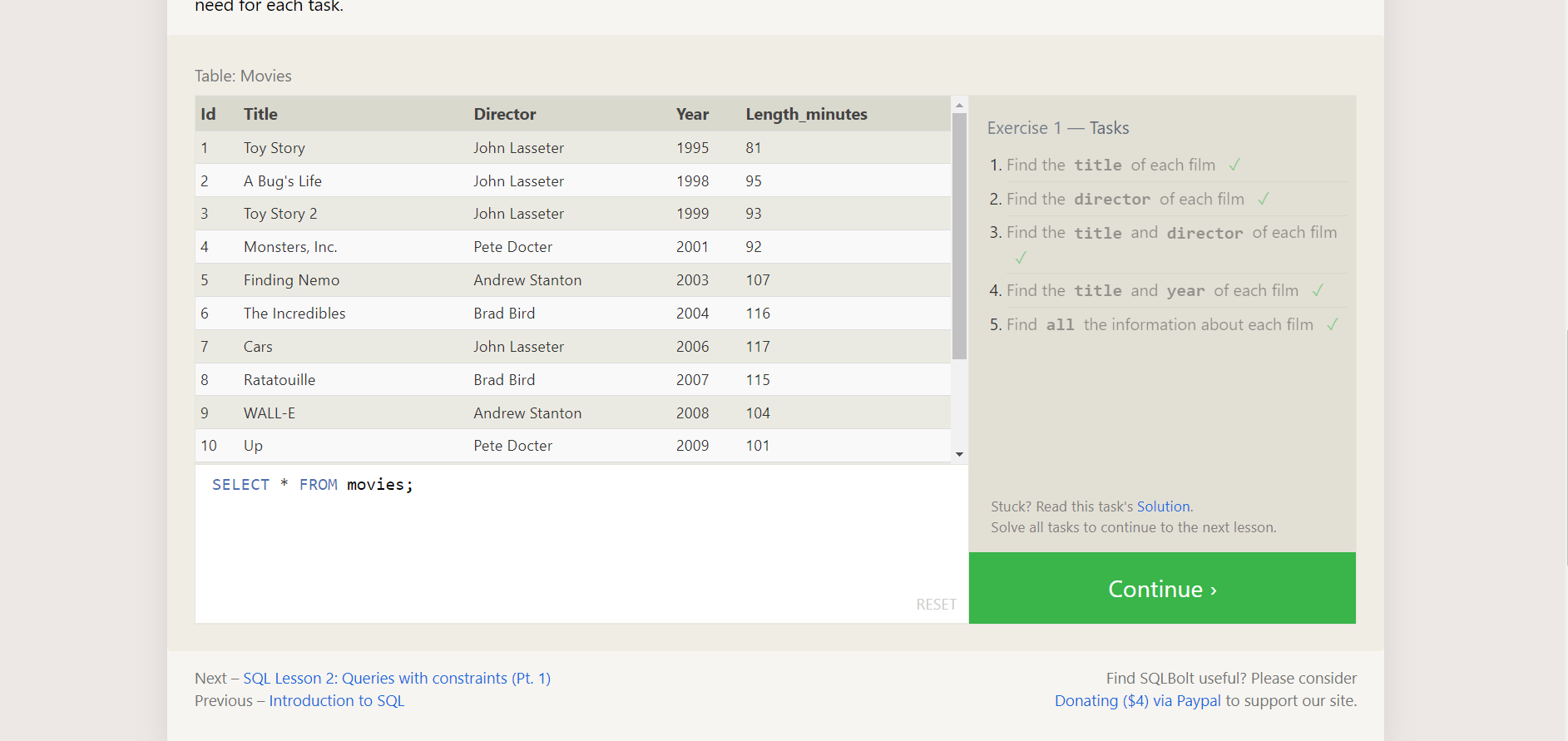
**My SQL -task1**

**SQL Lesson 1: SELECT queries 101**

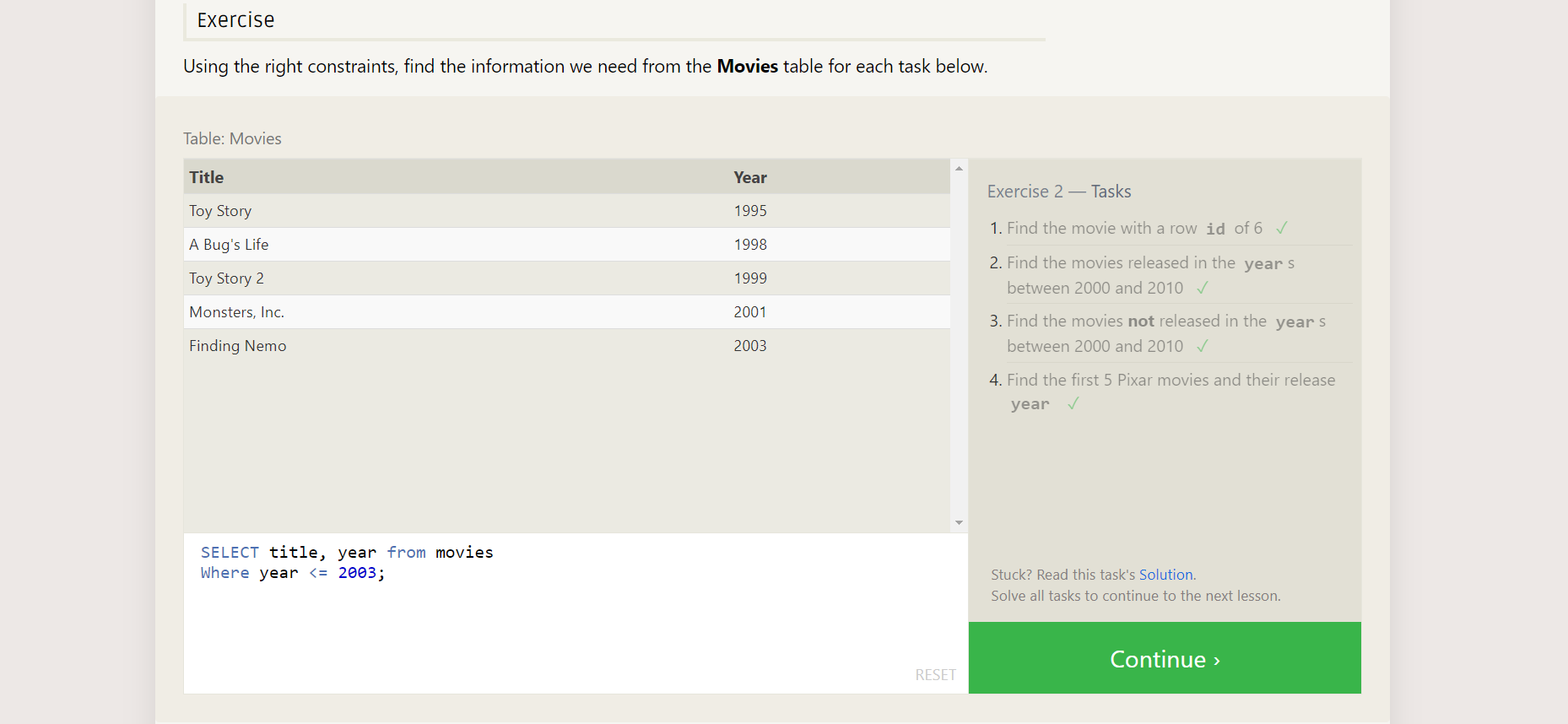
1. Select Title from movies;
2. Select Director from movies;
3. Select Title,Director from movies;
4. Select title,year from movies;
5. Select \* from movies;





**SQL Lesson 2: Queries with constraints (Pt. 1)**



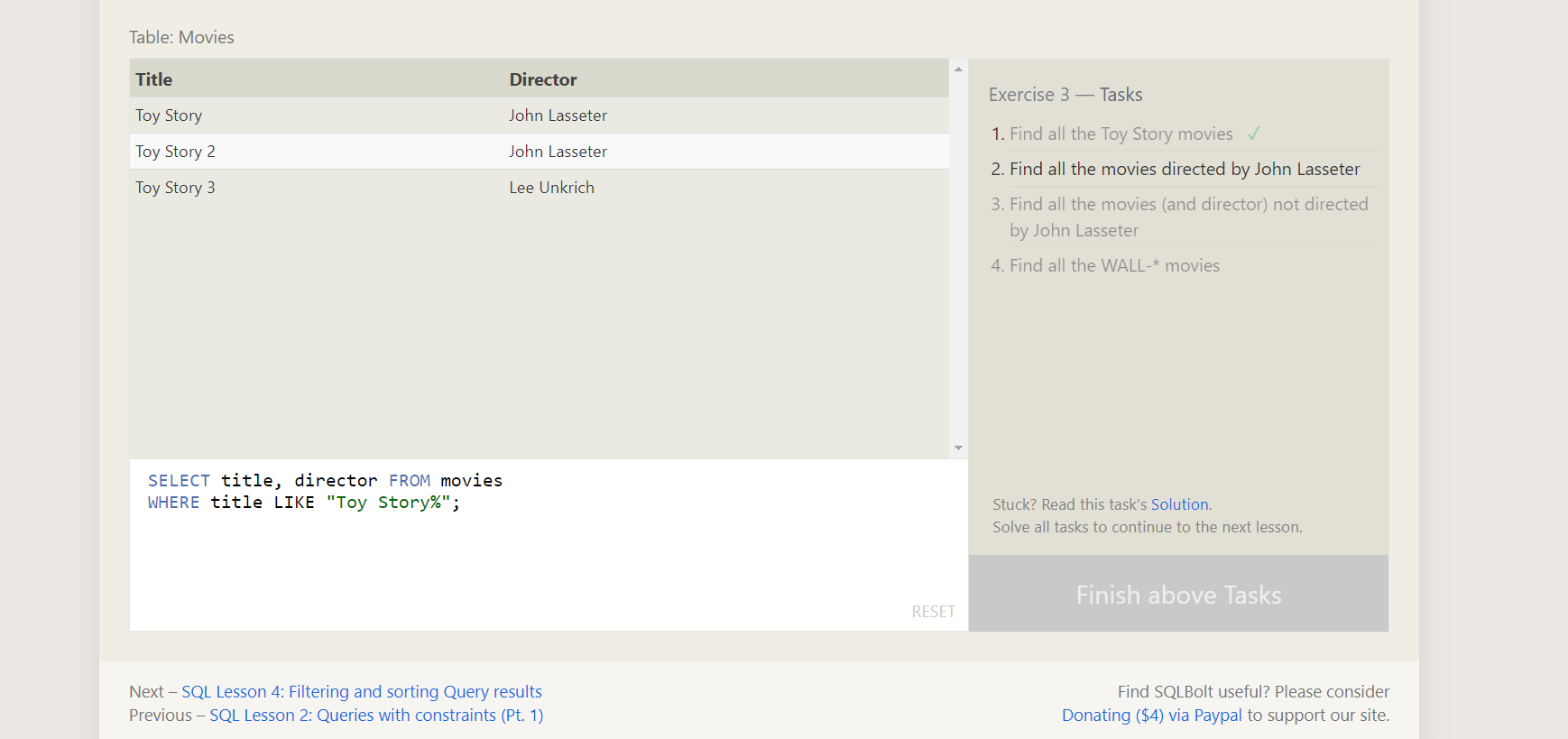


**SQL Lesson 3: Queries with constraints (Pt. 2)**

Ex3:

1. SELECT title, director FROM movies

WHERE title LIKE "Toy Story%";



2. SELECT title FROM movies

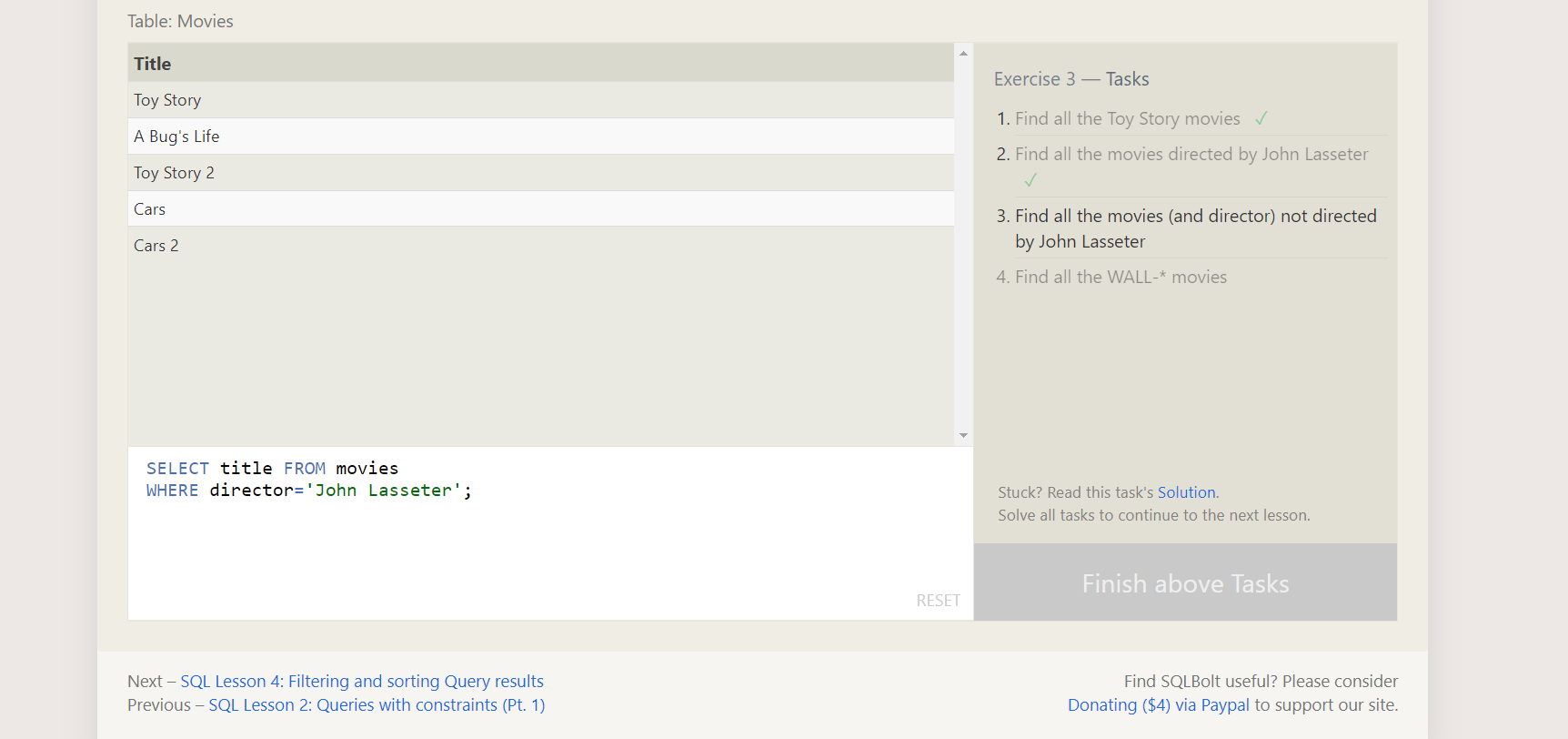
WHERE director='John Lasseter';

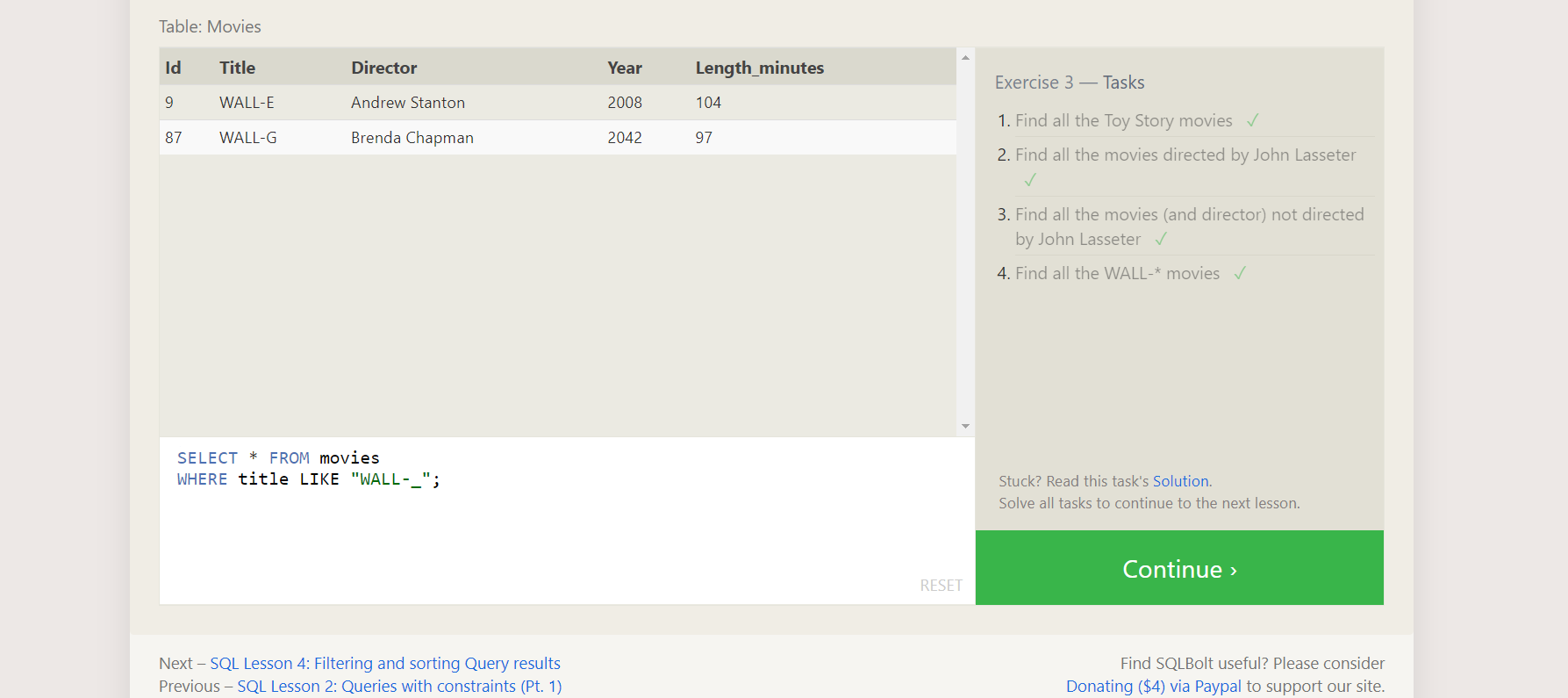
3. SELECT title FROM movies

WHERE not like director='John Lasseter';

4. SELECT \* FROM movies

WHERE title LIKE "WALL-\_";





**SQL Lesson 4: Filtering and sorting Query results**

1. SELECT DISTINCT director FROM movies

ORDER BY director ASC;

2. SELECT title,year FROM movies

ORDER BY year desc

LIMIT 4;

3. SELECT title FROM movies

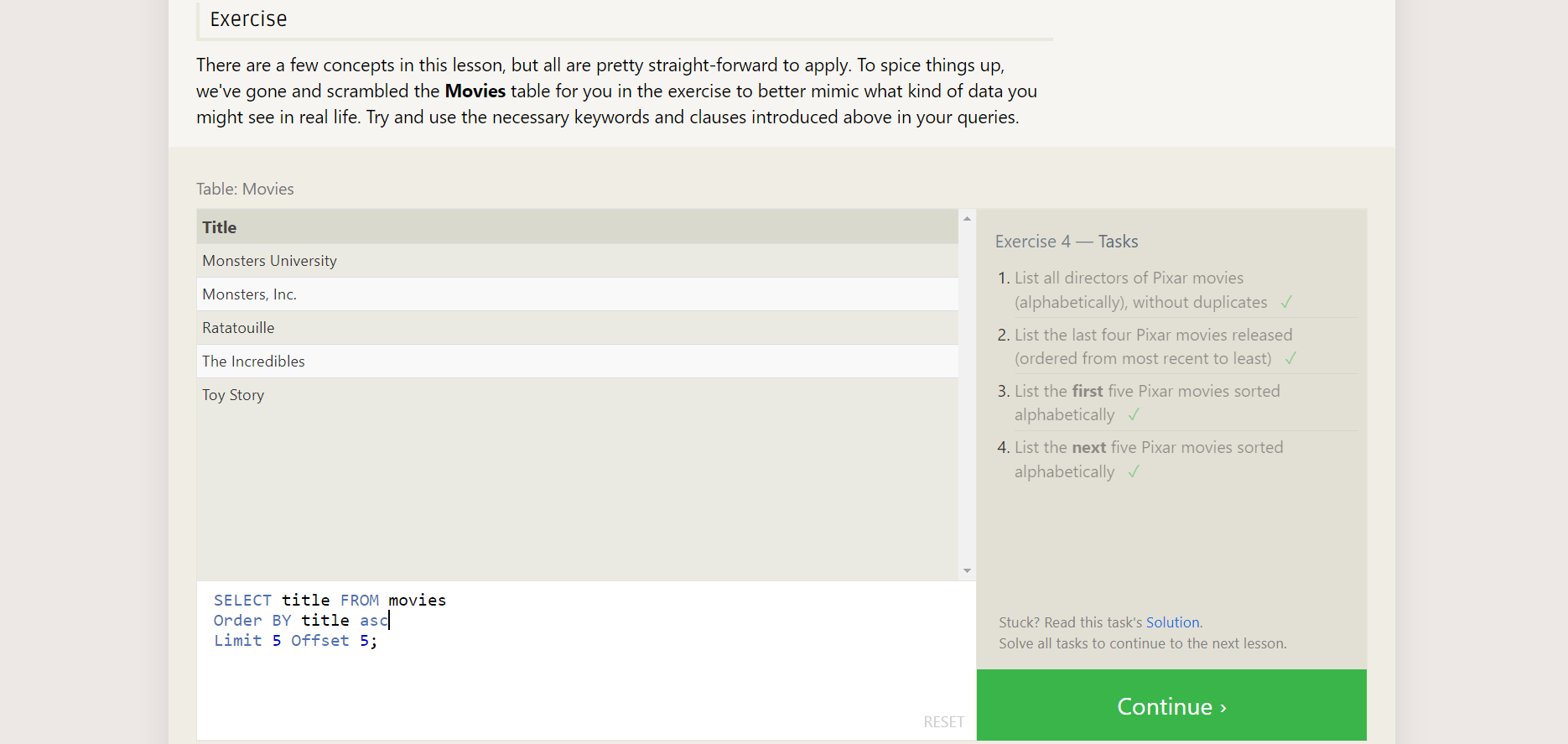
ORDER BY title asc

LIMIT 5;

4. SELECT title FROM movies

Order BY title asc

Limit 5 Offset 5;



**SQL Review: Simple SELECT Queries**

1.

2. SELECT city, latitude FROM north\_american\_cities

WHERE country = "United States"

ORDER BY latitude DESC;

3. SELECT city, longitude FROM north\_american\_cities

WHERE longitude < -87.629798

ORDER BY longitude ASC;

4. SELECT city, population FROM north\_american\_cities

WHERE country LIKE "Mexico"

ORDER BY population DESC

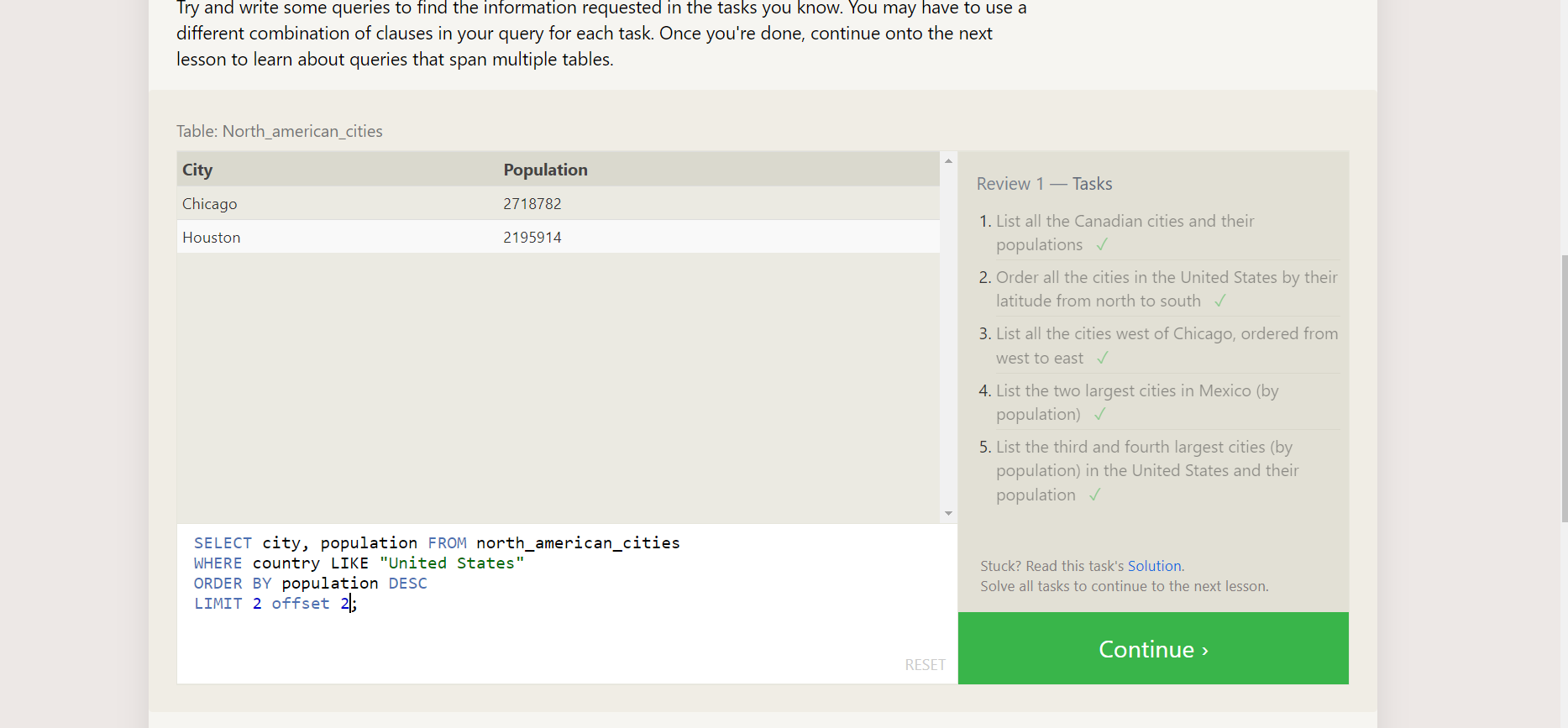
LIMIT 2;

5. SELECT city, population FROM north\_american\_cities

WHERE country LIKE "United States"

ORDER BY population DESC

LIMIT 2 offset 2;



**SQL Lesson 6: Multi-table queries with JOINs**

1. SELECT title, domestic\_sales, international\_sales

FROM movies

JOIN boxoffice

ON movies.id = boxoffice.movie\_id;

2. SELECT title, domestic\_sales, international\_sales

FROM movies

JOIN boxoffice

ON movies.id = boxoffice.movie\_id

WHERE international\_sales > domestic\_sales;

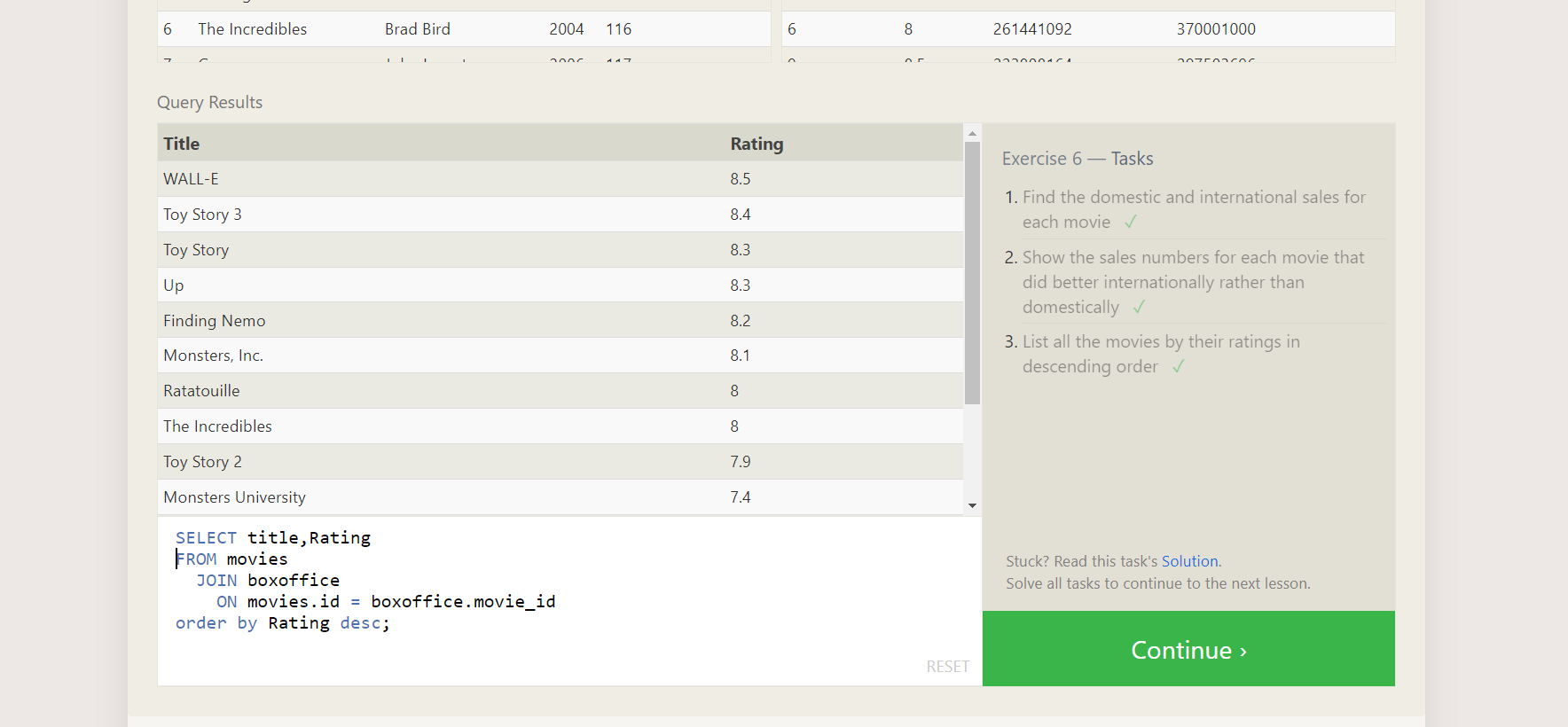
3. SELECT title,Rating

FROM movies

JOIN boxoffice

ON movies.id = boxoffice.movie\_id

order by Rating desc;



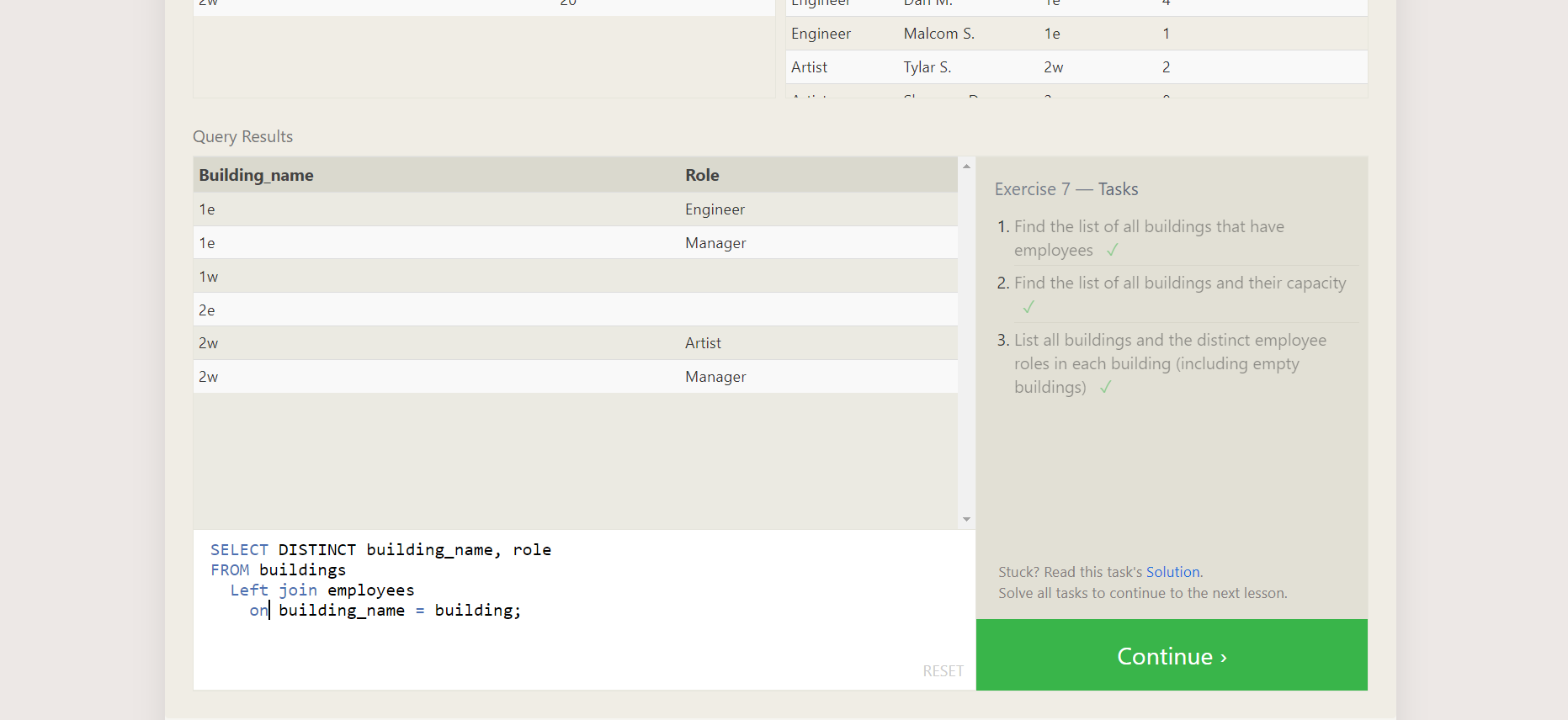
**SQL Lesson 7: OUTER JOINs**

1. SELECT DISTINCT building FROM employees;
2. SELECT \* FROM buildings;
3. SELECT DISTINCT building\_name, role

FROM buildings

Left join employees

on building\_name = building;



**SQL Lesson 8: A short note on NULLs**

1. SELECT name, role FROM employees

WHERE building IS NULL;

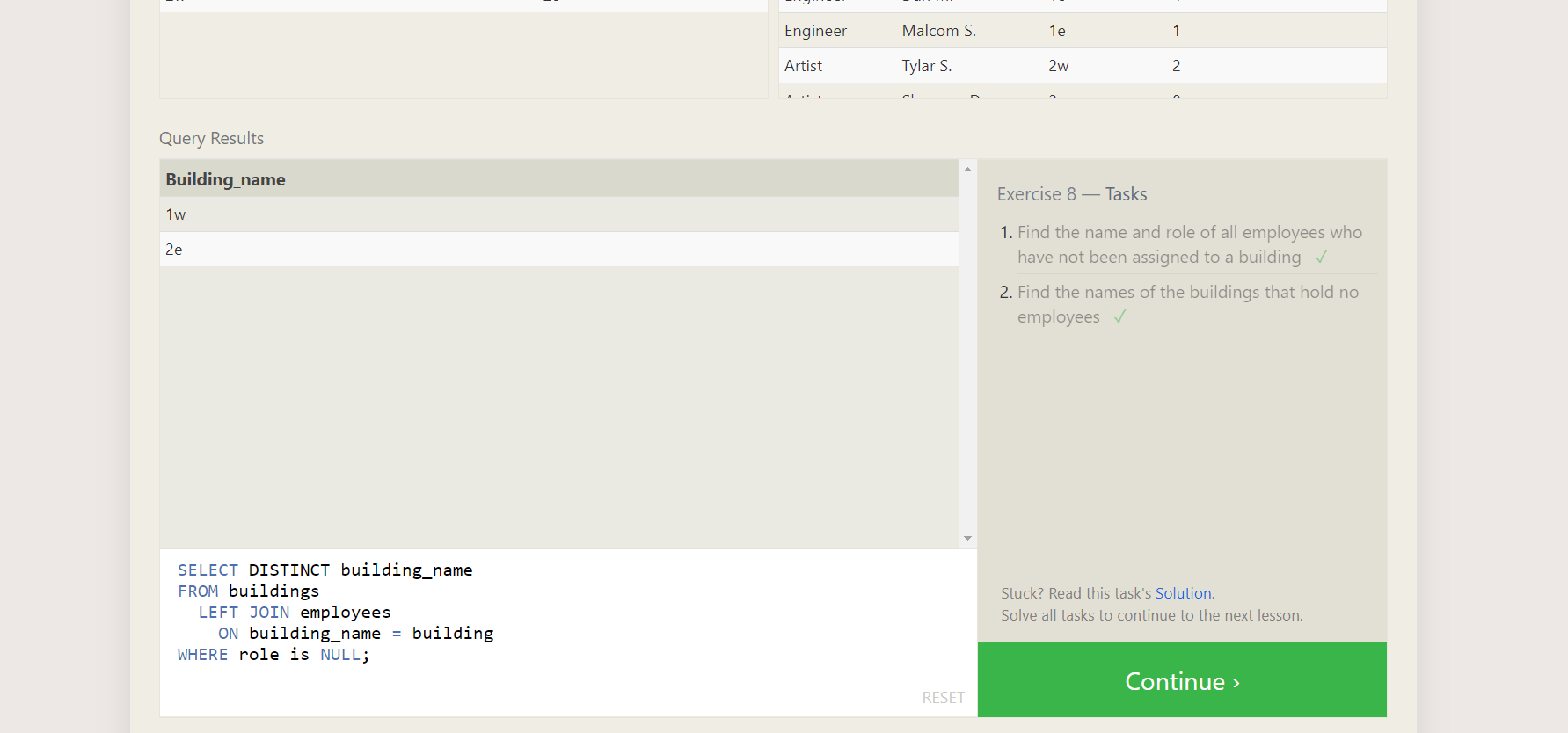
2. SELECT DISTINCT building\_name

FROM buildings

LEFT JOIN employees

ON building\_name = building

WHERE role is NULL;



**SQL Lesson 9: Queries with expressions**

1. SELECT title, (domestic\_sales + international\_sales) / 1000000 AS gross\_sales\_millions

FROM movies

JOIN boxoffice

ON movies.id = boxoffice.movie\_id;

2. SELECT title, rating \* 10 AS rating\_percent

FROM movies

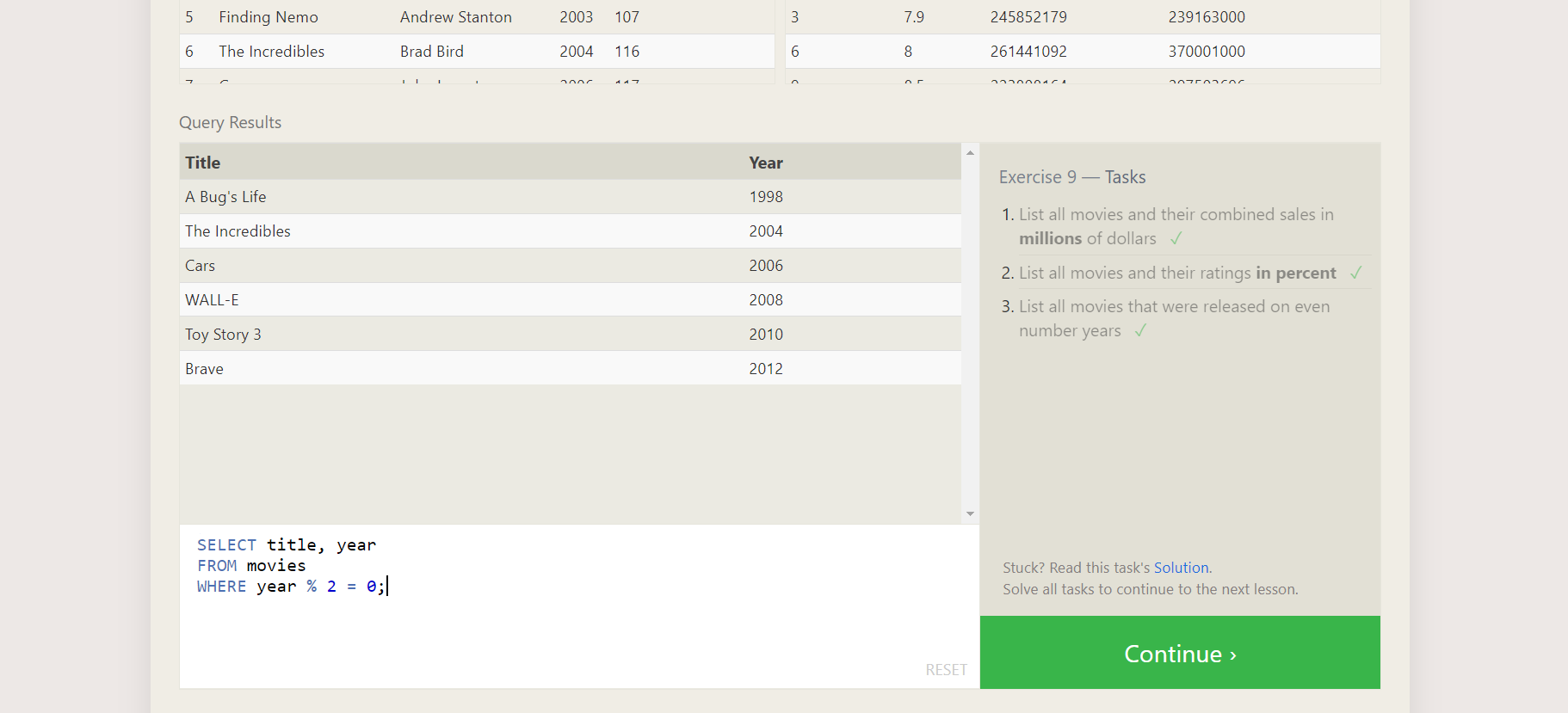
JOIN boxoffice

ON movies.id = boxoffice.movie\_id;

3. SELECT title, year

FROM movies

WHERE year % 2 = 0;



**SQL Lesson 10: Queries with aggregates (Pt. 1)**

1. SELECT MAX(years\_employed) as Max\_years\_employed

FROM employees;

2. SELECT role, AVG(years\_employed) as Average\_years\_employed

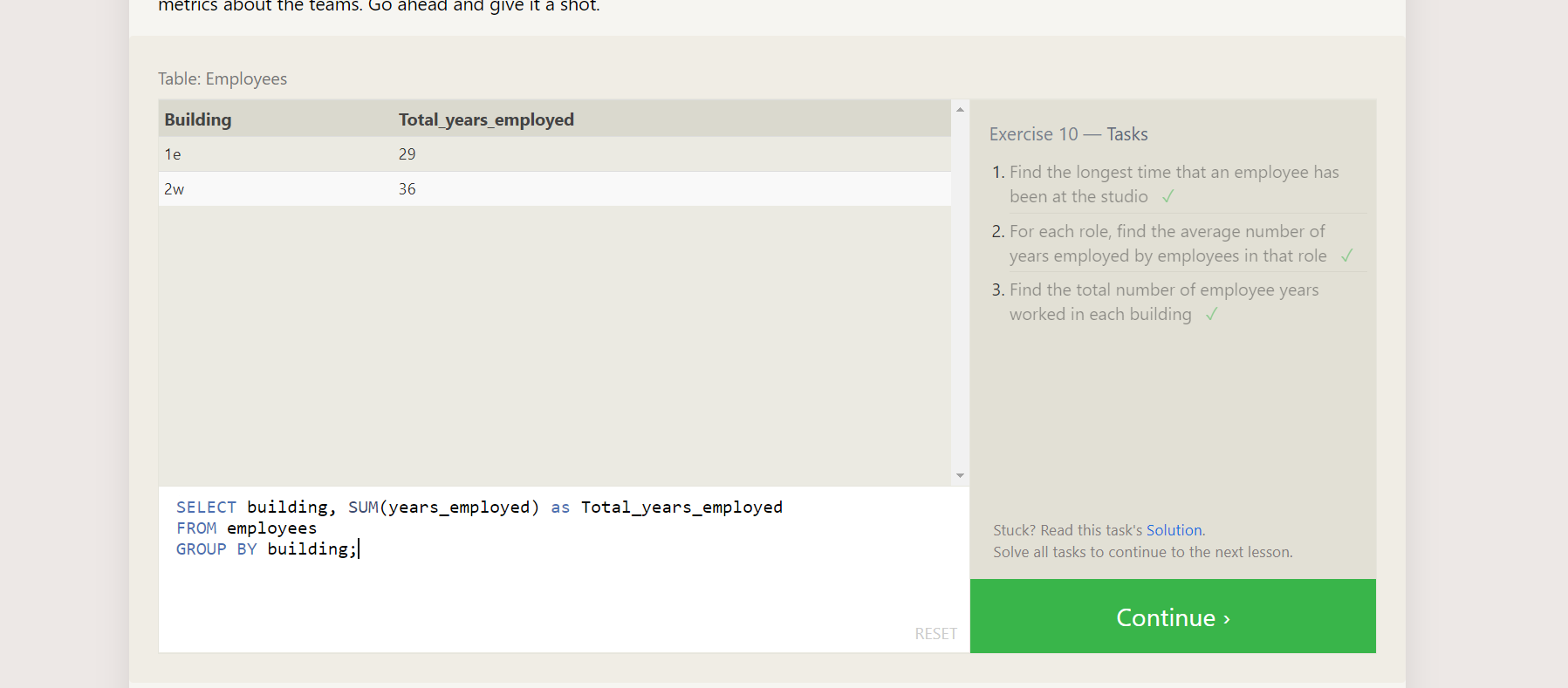
FROM employees

GROUP BY role;

3. SELECT building, SUM(years\_employed) as Total\_years\_employed

FROM employees

GROUP BY building;



**SQL Lesson 11: Queries with aggregates (Pt. 2)**

1. SELECT role, COUNT(\*) as Number\_of\_artists

FROM employees

WHERE role = "Artist";

2. SELECT role, COUNT(\*) as Number\_of\_artists

FROM employees

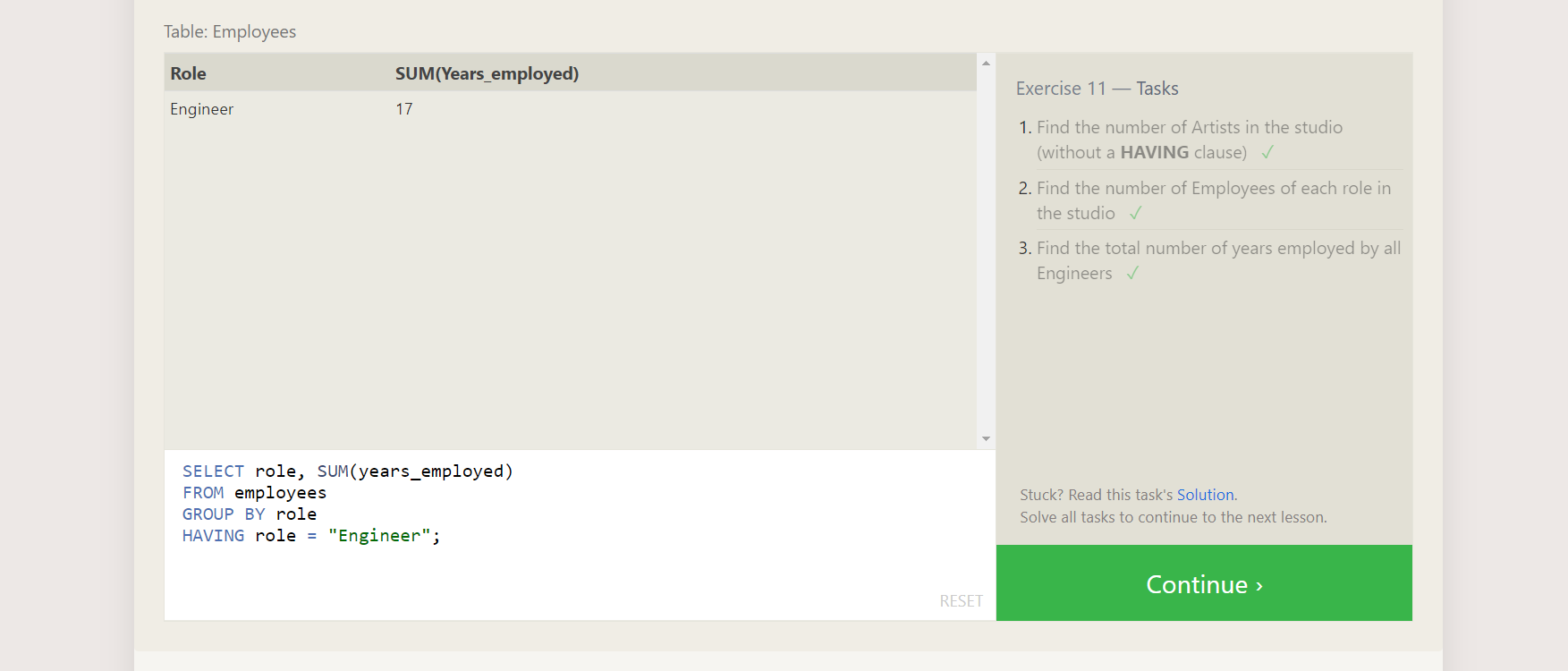
WHERE role = "Artist";

3. SELECT role, SUM(years\_employed)

FROM employees

GROUP BY role

HAVING role = "Engineer";



**SQL Lesson 12: Order of execution of a Query**

1. SELECT director, COUNT(id) as Num\_movies\_directed

FROM movies

GROUP BY director;

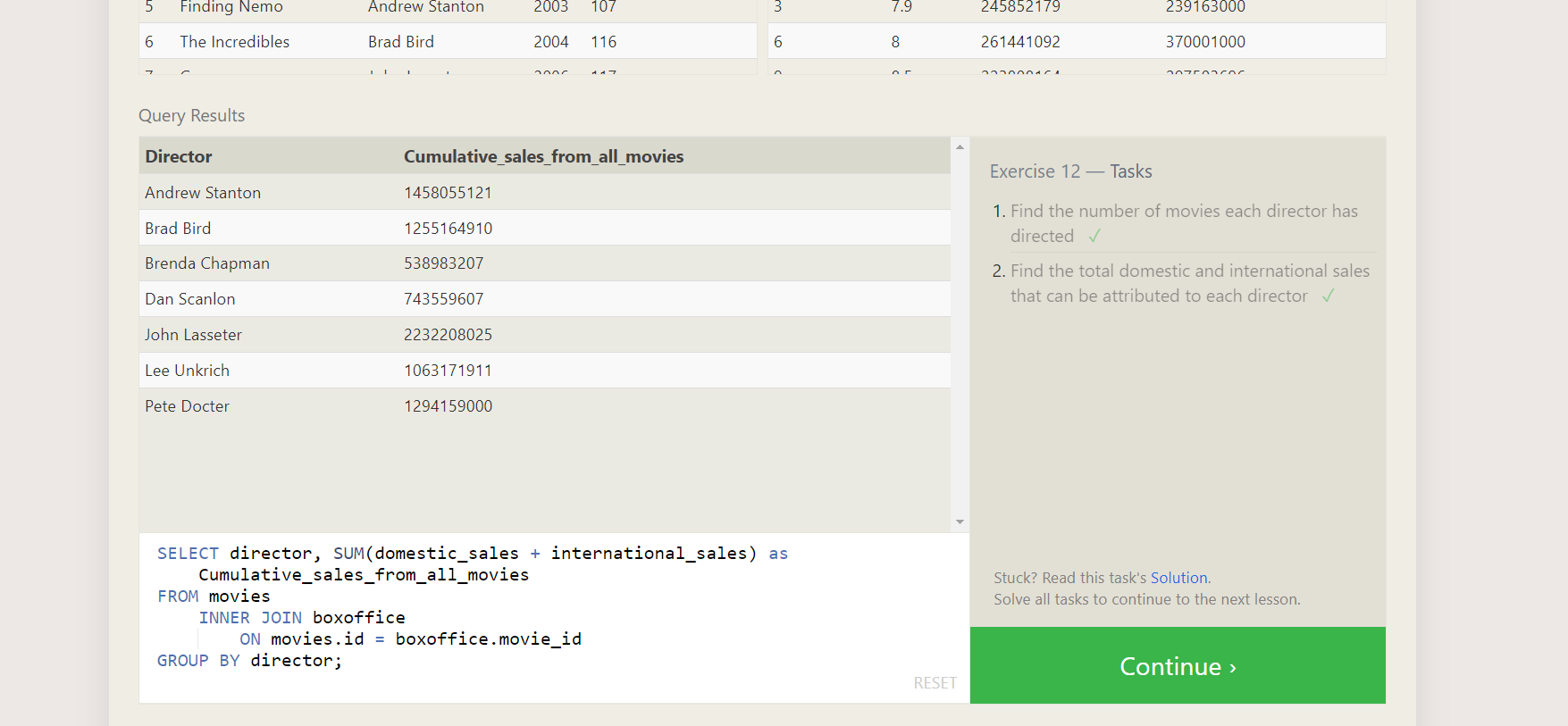
2. SELECT director, SUM(domestic\_sales + international\_sales) as Cumulative\_sales\_from\_all\_movies

FROM movies

INNER JOIN boxoffice

ON movies.id = boxoffice.movie\_id

GROUP BY director;



**SQL Lesson 13: Inserting rows**

1. INSERT INTO movies VALUES (4, "Toy Story 4", "Brad Bird", 2015, 90);
2. INSERT INTO boxoffice VALUES (4, 8.7, 340000000, 270000000);



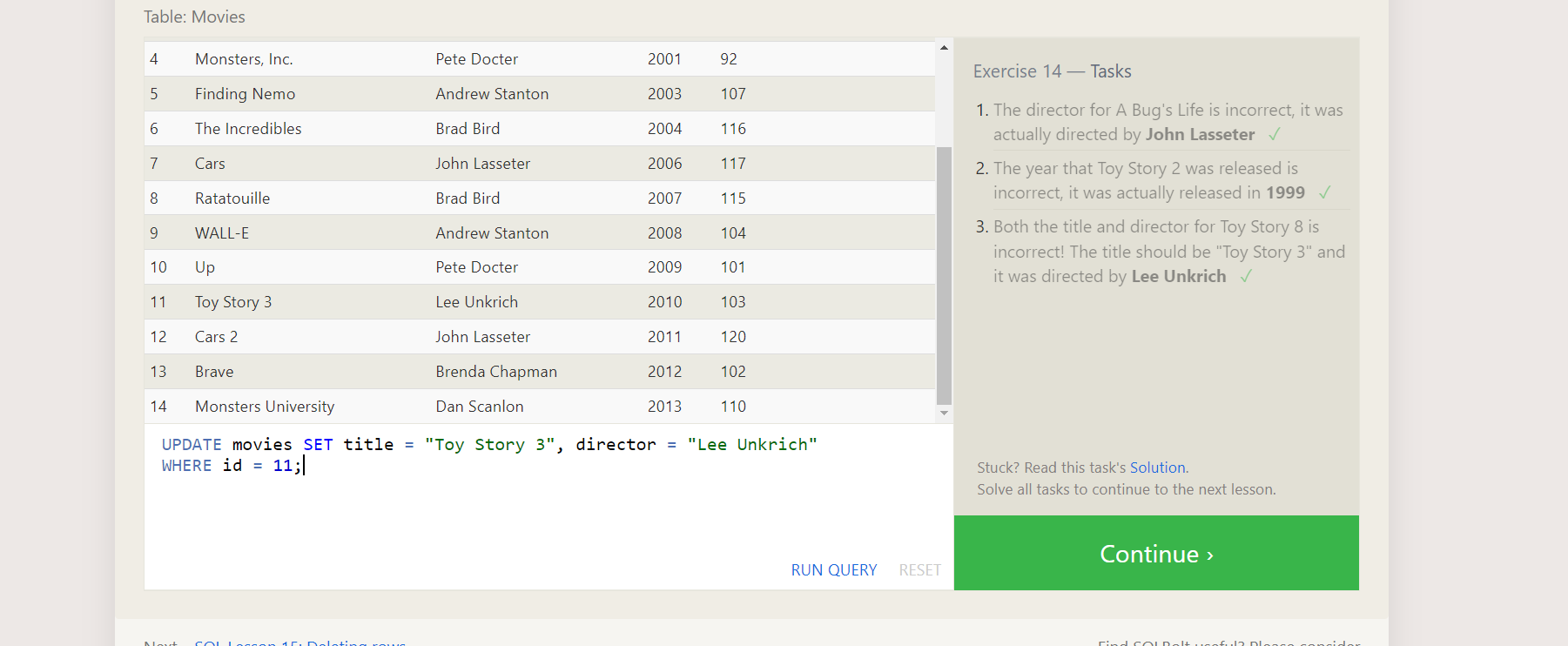
**SQL Lesson 14: Updating rows**

1. update Movies set director='John Lasseter' where title="A Bug's Life"
2. UPDATE movies

SET year = 1999

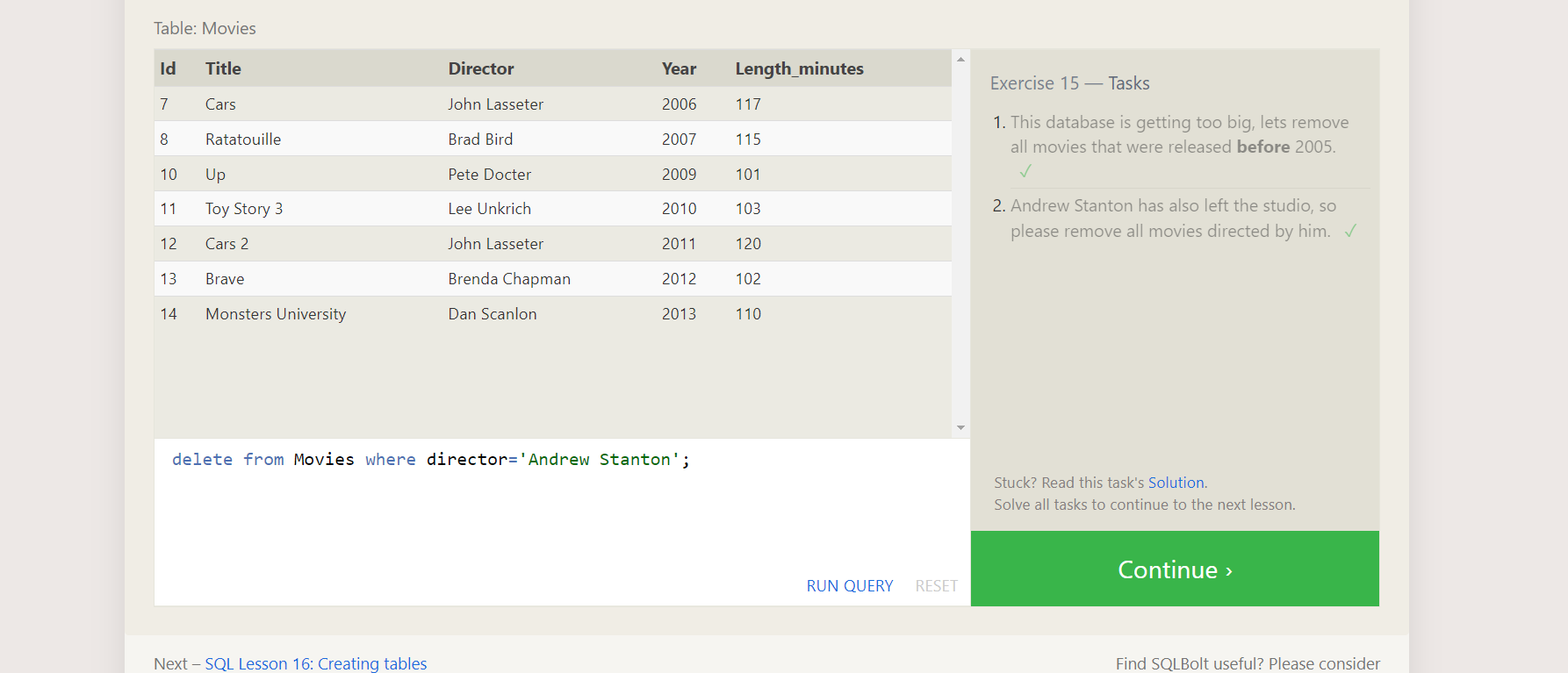
WHERE id = 3;

1. UPDATE movies SET title = "Toy Story 3", director = "Lee Unkrich" WHERE id = 11;

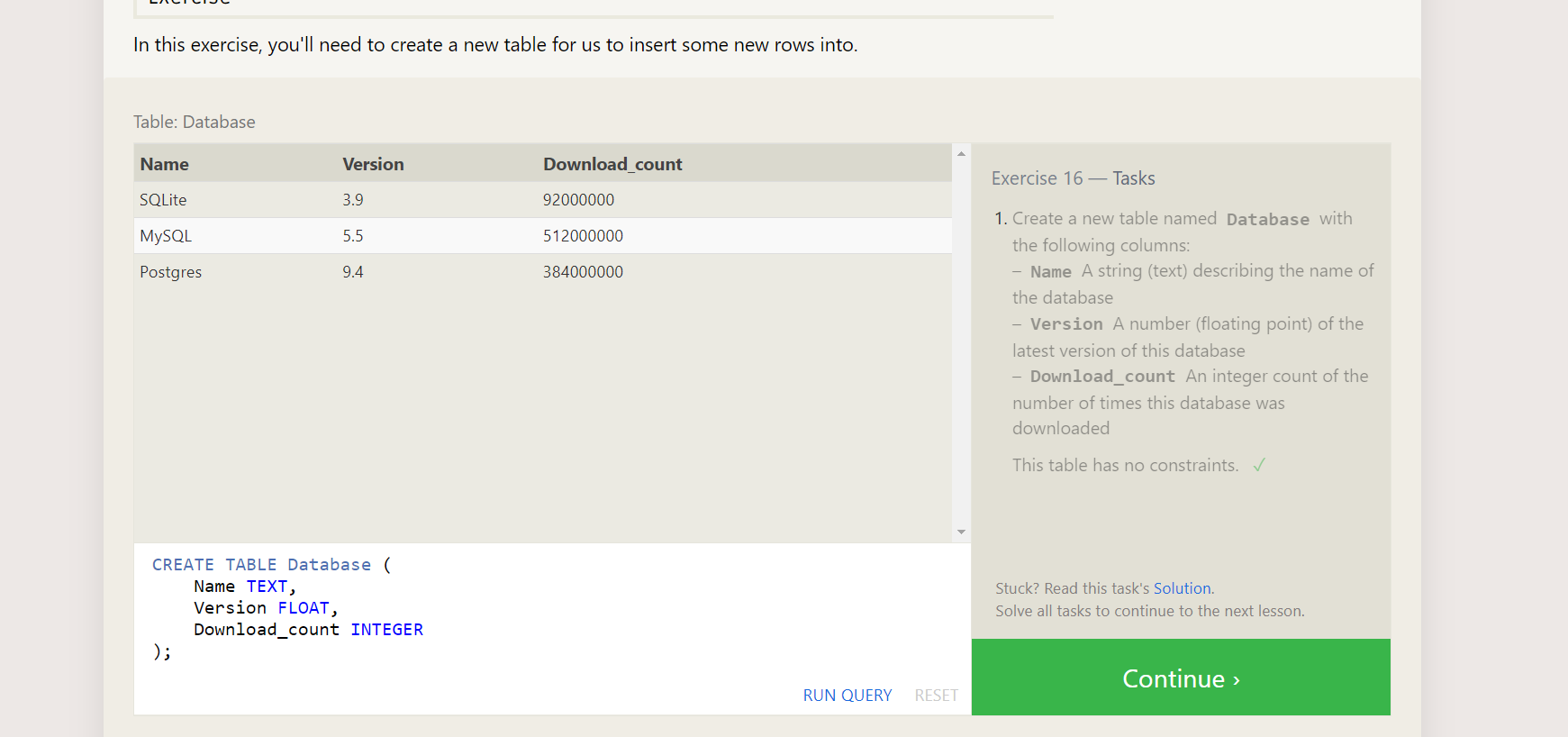


**SQL Lesson 15: Deleting rows**

1. delete from Movies where year<2005;
2. delete from Movies where director='Andrew Stanton';



**SQL Lesson 16: Creating tables**



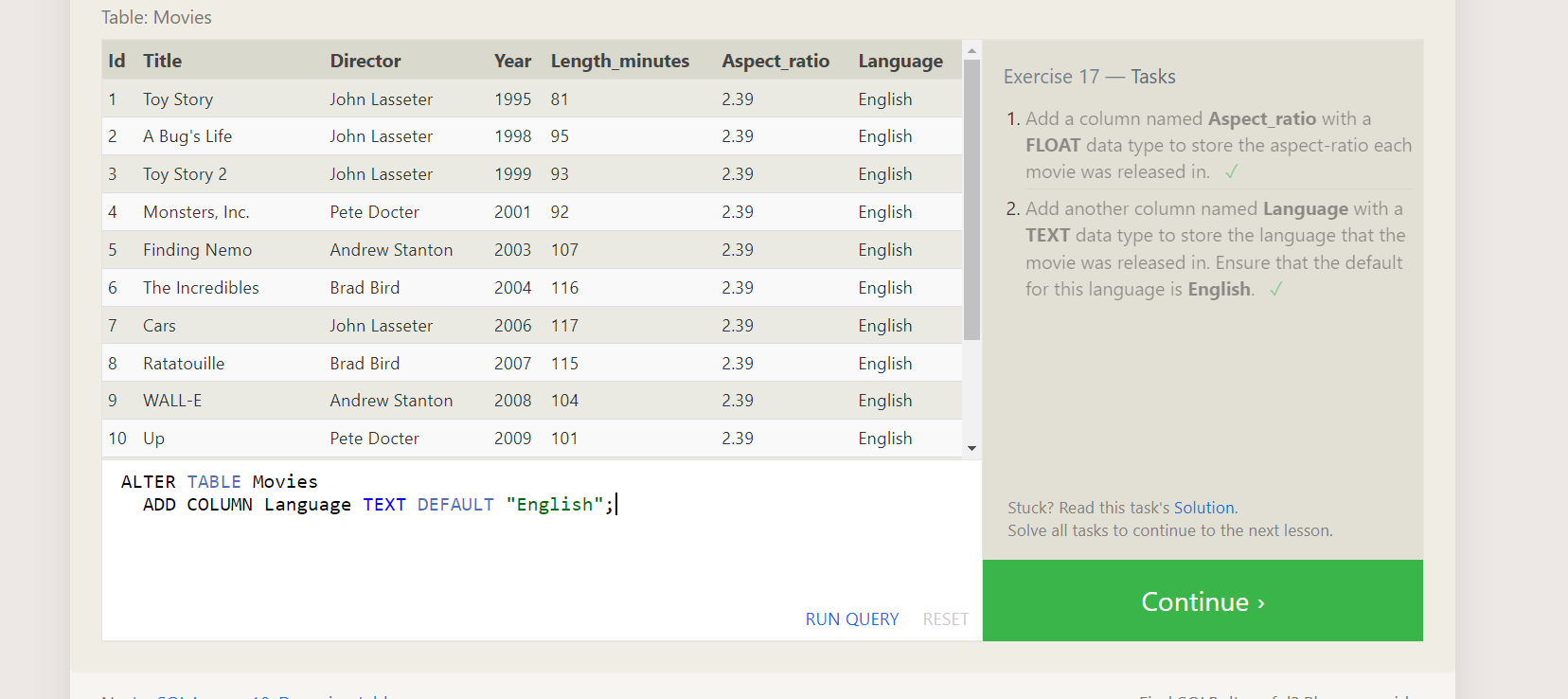
**SQL Lesson 17: Altering tables**

1. ALTER TABLE Movies

ADD COLUMN Aspect\_ratio FLOAT DEFAULT 2.39;

2. ALTER TABLE Movies

ADD COLUMN Language TEXT DEFAULT "English";



**SQL Lesson 18: Dropping tables**

1.DROP TABLE IF EXISTS Movies;

2. DROP TABLE IF EXISTS Boxoffice;

